Modulo 2 Adition (XOR)

+	0	1
0	0	1
1	1	0

Modulo 2 Multiplication (XOR)

*	0	1
0	0	0
1	0	1

Figure 1

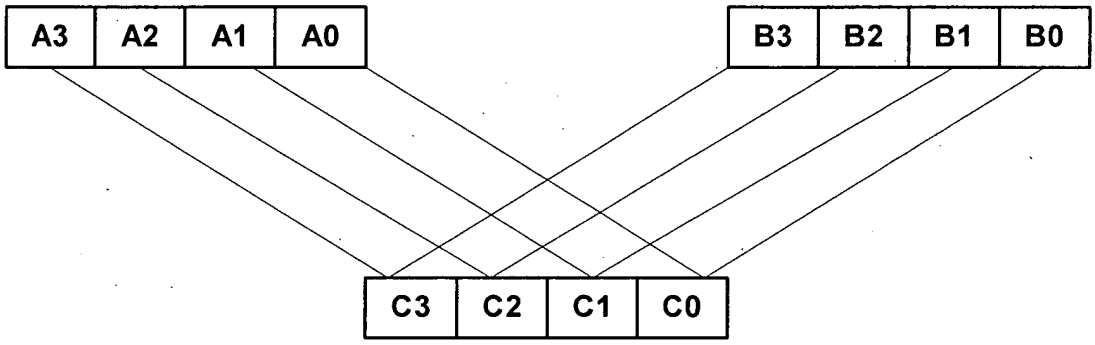


Figure 2

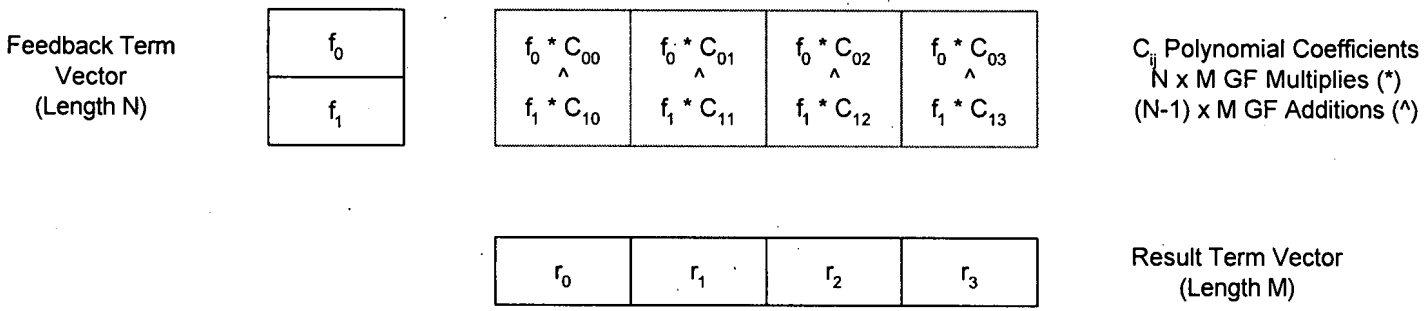

Feedback Term Vector (Length N)
f_0
f_1
$f_0 * C_{00}$ ^ $f_1 * C_{10}$
$f_0 * C_{01}$ ^ $f_1 * C_{11}$
$f_0 * C_{02}$ ^ $f_1 * C_{12}$
$f_0 * C_{03}$ ^ $f_1 * C_{13}$
C_{ij} Polynomial Coefficients
N x M GF Multiplies (*)
(N-1) x M GF Additions (^)
r_0
r_1
r_2
r_3
Result Term Vector
(Length M)

Figure 3

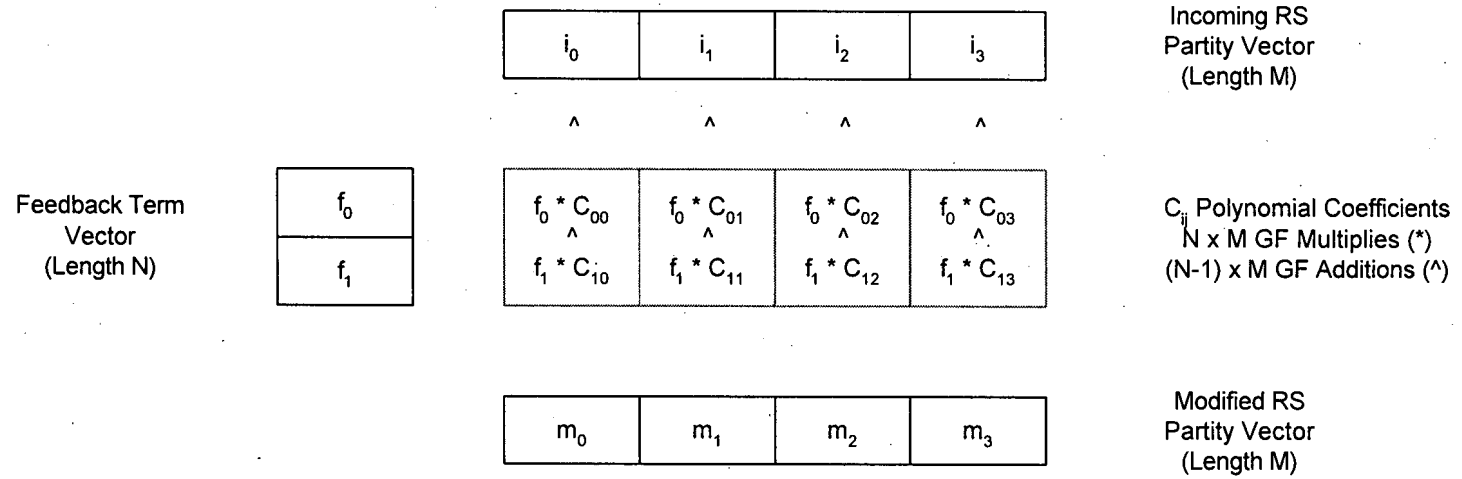

Incoming RS
Partity Vector
(Length M)
i0
i1
i2
i3
^
Feedback Term
Vector
(Length N)
f0
f1
f0 * C00
^
f1 * C10
f0 * C01
^
f1 * C11
f0 * C02
^
f1 * C12
f0 * C03
^
f1 * C13
Cij Polynomial Coefficients
N x M GF Multiplies (*)
(N-1) x M GF Additions (^)
Modified RS
Partity Vector
(Length M)
m0
m1
m2
m3

Figure 4

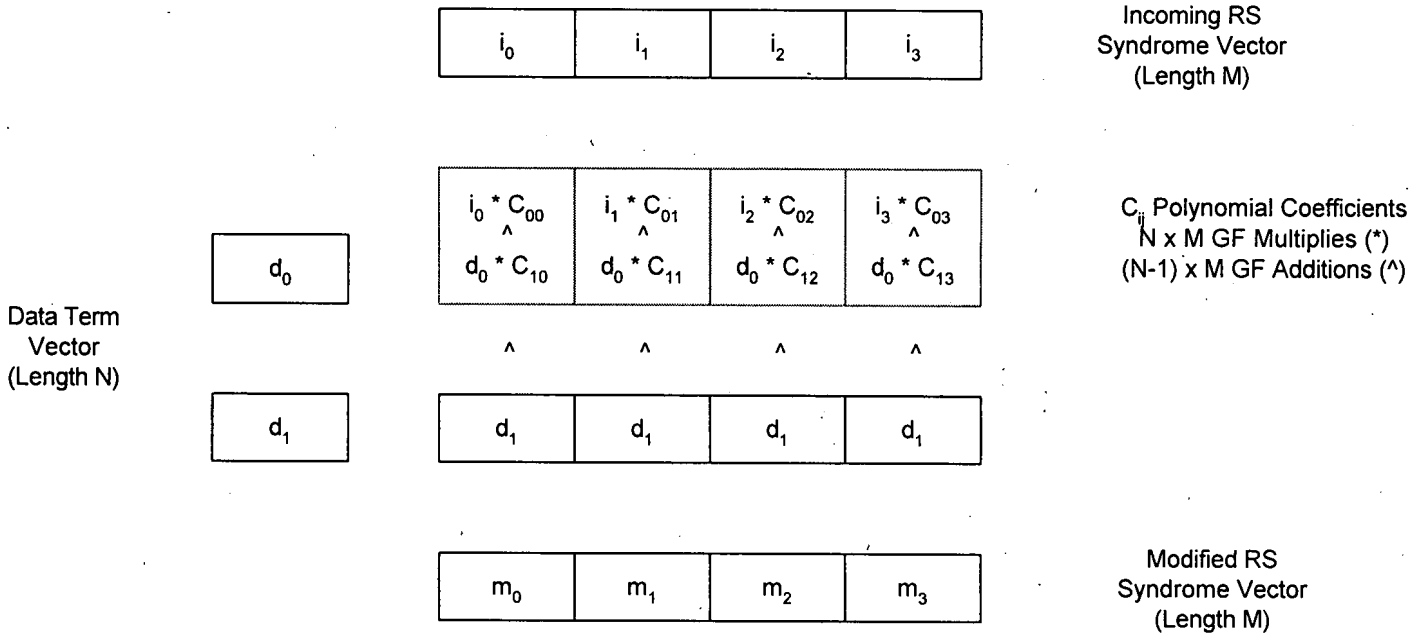
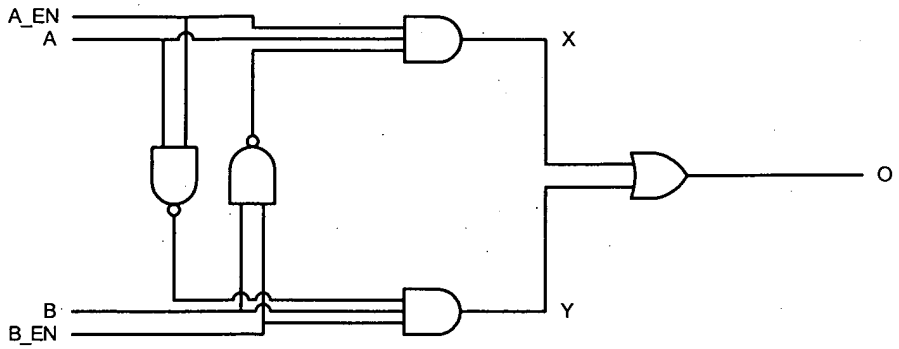
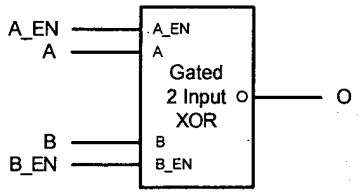

Incoming RS
Syndrome Vector
(Length M)
i_0 i_1 i_2 i_3
C_{ij} Polynomial Coefficients
N x M GF Multiplies (*)
(N-1) x M GF Additions (^)
$i_0 * C_{00}$ ^ $d_0 * C_{10}$
$i_1 * C_{01}$ ^ $d_0 * C_{11}$
$i_2 * C_{02}$ ^ $d_0 * C_{12}$
$i_3 * C_{03}$ ^ $d_0 * C_{13}$
Data Term
Vector
(Length N)
d_0
d_1
d_1 d_1 d_1 d_1
Modified RS
Syndrome Vector
(Length M)
m_0 m_1 m_2 m_3

Figure 5



Gated 2 Input XOR Logic



Gated 2 Input XOR Symbol

A	B	A_EN	B_EN	X	Y	O	Notes
-	-	0	0	0	0	0	Block
0	0	1	0	0	0	0	
0	1	1	0	0	0	0	Pass A
1	0	1	0	1	0	1	
1	1	1	0	1	0	1	
0	0	0	1	0	0	0	
0	1	0	1	0	1	1	Pass B
1	0	0	1	0	0	0	
1	1	0	1	0	1	1	
0	0	1	1	0	0	0	
0	1	1	1	0	1	1	A ^ B
1	0	1	1	1	0	1	
1	1	1	1	0	0	0	

Gated 2 Input XOR Truth Table

Figure 6

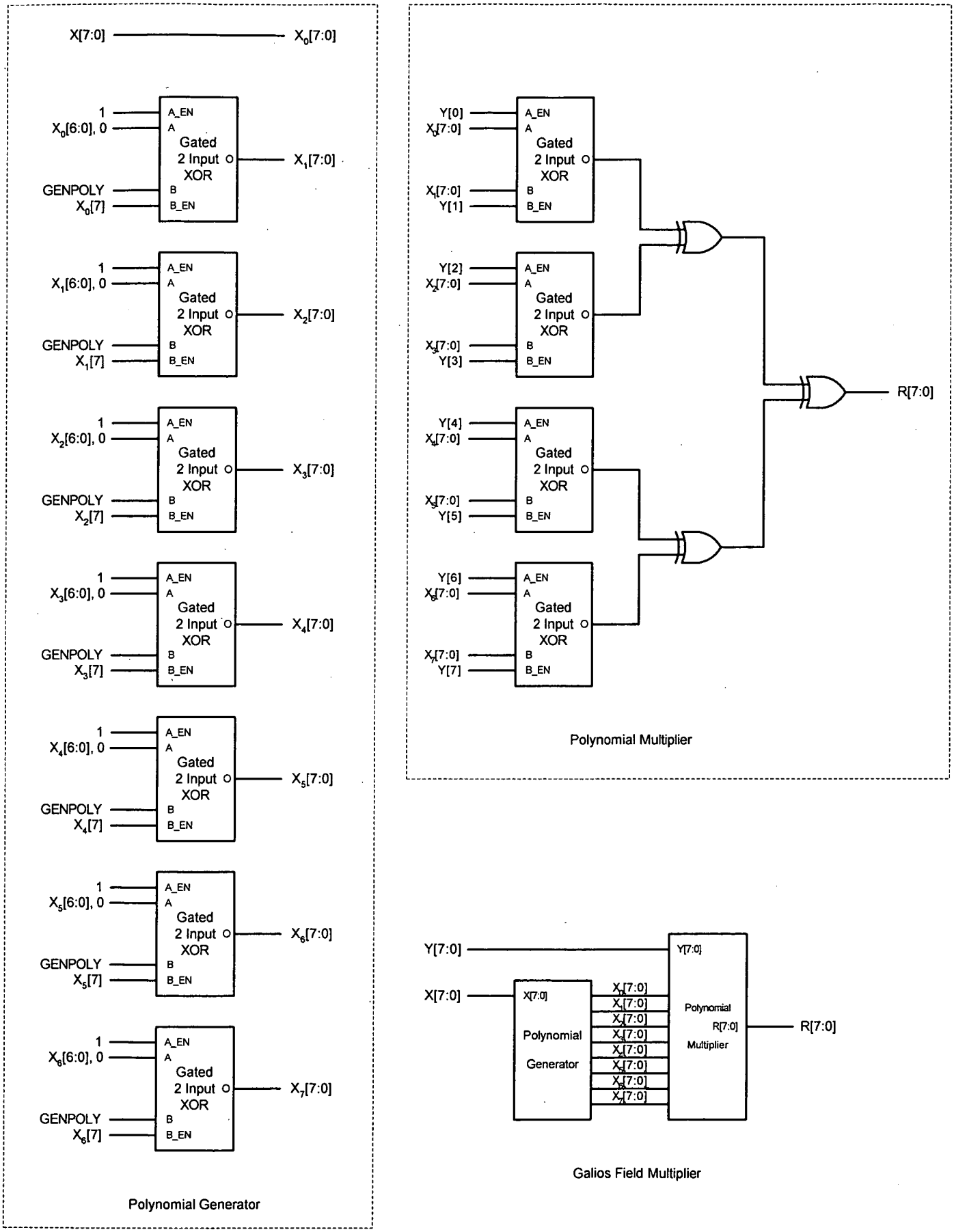

X[7:0]
X0[7:0]
1
X0[6:0], 0
GENPOLY
X0[7]
A_EN
A
Gated
2 Input
XOR
B
B_EN
X1[7:0]
X1[6:0], 0
X1[7]
X2[7:0]
X2[6:0], 0
X2[7]
X3[7:0]
X3[6:0], 0
X3[7]
X4[7:0]
X4[6:0], 0
X4[7]
X5[7:0]
X5[6:0], 0
X5[7]
X6[7:0]
X6[6:0], 0
X6[7]
X7[7:0]
Polynomial Generator
Y[0]
X0[7:0]
X1[7:0]
Y[1]
Y[2]
X2[7:0]
X3[7:0]
Y[3]
Y[4]
X4[7:0]
X5[7:0]
Y[5]
Y[6]
X6[7:0]
X7[7:0]
Y[7]
R[7:0]
Polynomial Multiplier
Y[7:0]
X[7:0]
Polynomial
Generator
Polynomial
R[7:0]
Multiplier
Galios Field Multiplier

Figure 7

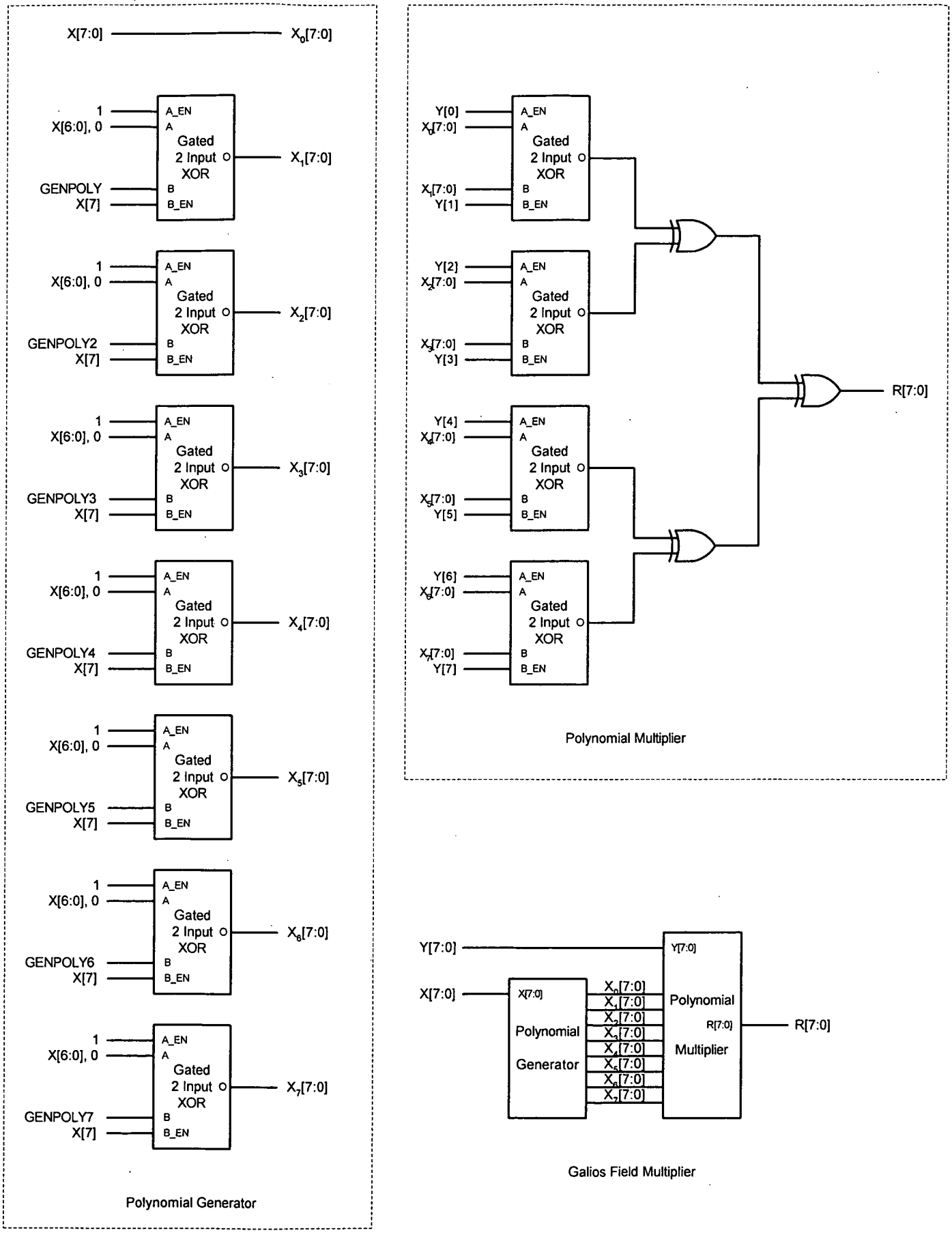
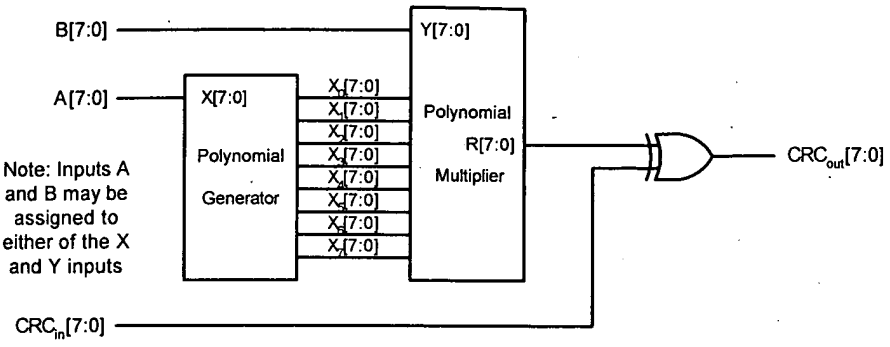

X[7:0]
X0[7:0]
1
X[6:0], 0
GENPOLY
X[7]
A_EN
A
Gated
2 Input
XOR
B
B_EN
X1[7:0]
GENPOLY2
X2[7:0]
GENPOLY3
X3[7:0]
GENPOLY4
X4[7:0]
GENPOLY5
X5[7:0]
GENPOLY6
X6[7:0]
GENPOLY7
X7[7:0]
Polynomial Generator
Y[0]
X0[7:0]
X1[7:0]
Y[1]
Y[2]
X2[7:0]
X3[7:0]
Y[3]
Y[4]
X4[7:0]
X5[7:0]
Y[5]
Y[6]
X6[7:0]
X7[7:0]
Y[7]
R[7:0]
Polynomial Multiplier
Y[7:0]
X[7:0]
Polynomial
Generator
Polynomial
Multiplier
Galios Field Multiplier

Figure 8



Scalar instruction: crc = crc ^ gf_mult (a, b)

As used in the example software, a is the feedback term and b is the polynomial term

Figure 9

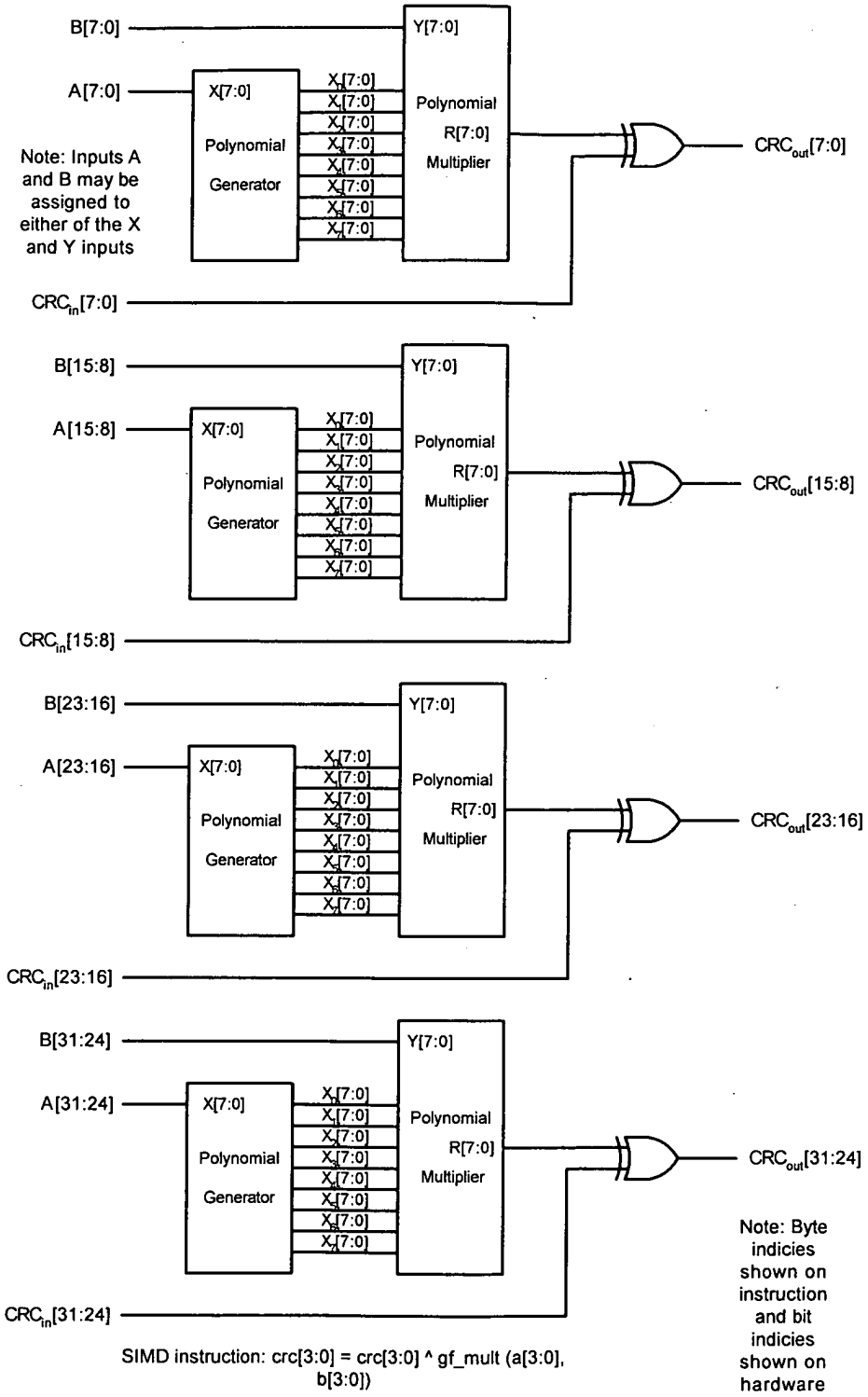
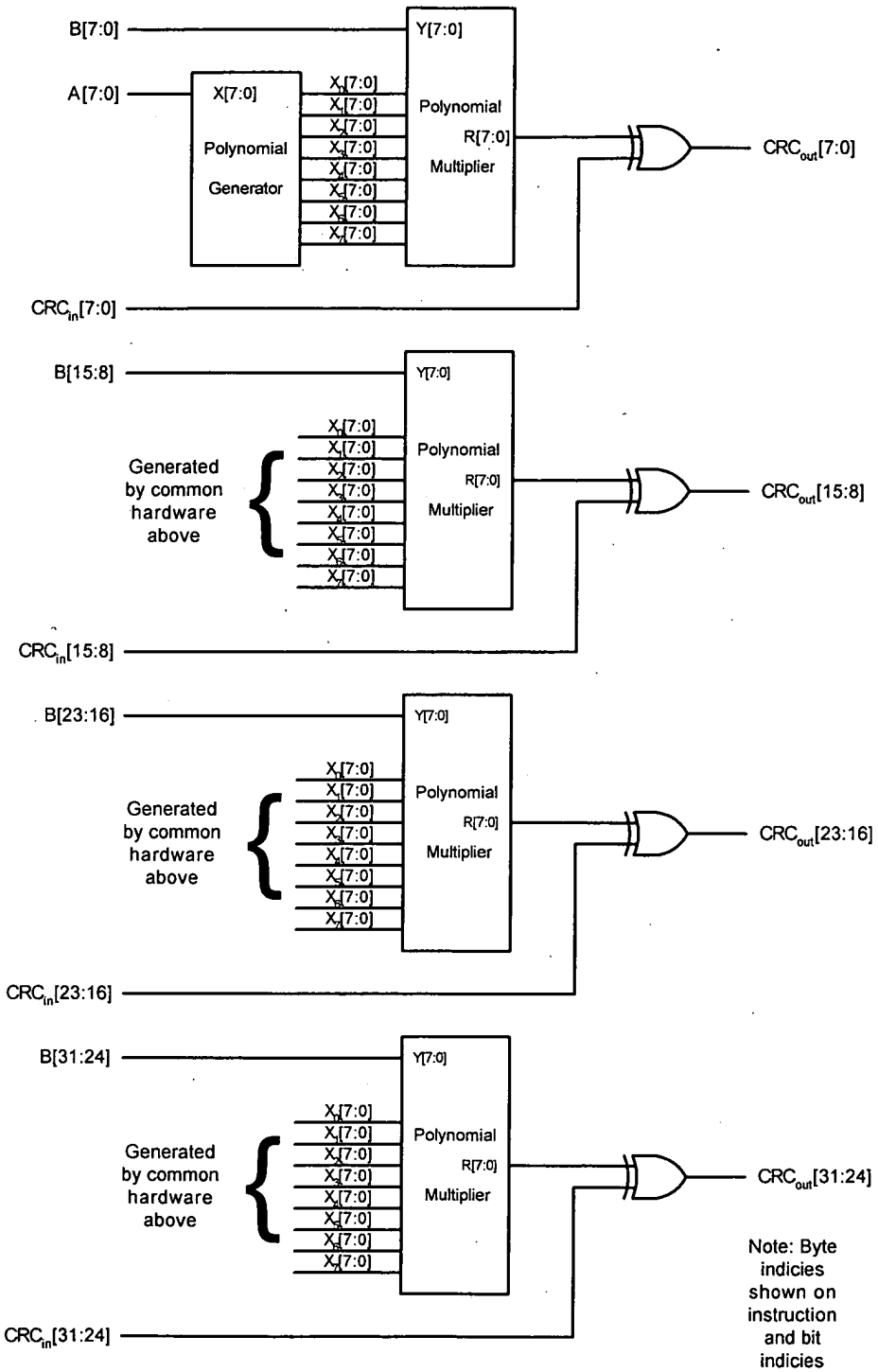

B[7:0]
A[7:0]
X[7:0]
Polynomial
Generator
Y[7:0]
Polynomial
R[7:0]
Multiplier
CRC_out[7:0]
Note: Inputs A and B may be assigned to either of the X and Y inputs
CRC_in[7:0]
B[15:8]
A[15:8]
CRC_out[15:8]
CRC_in[15:8]
B[23:16]
A[23:16]
CRC_out[23:16]
CRC_in[23:16]
B[31:24]
A[31:24]
CRC_out[31:24]
CRC_in[31:24]
SIMD instruction: crc[3:0] = crc[3:0] ^ gf_mult (a[3:0], b[3:0])
Note: Byte indicies shown on instruction and bit indicies shown on hardware

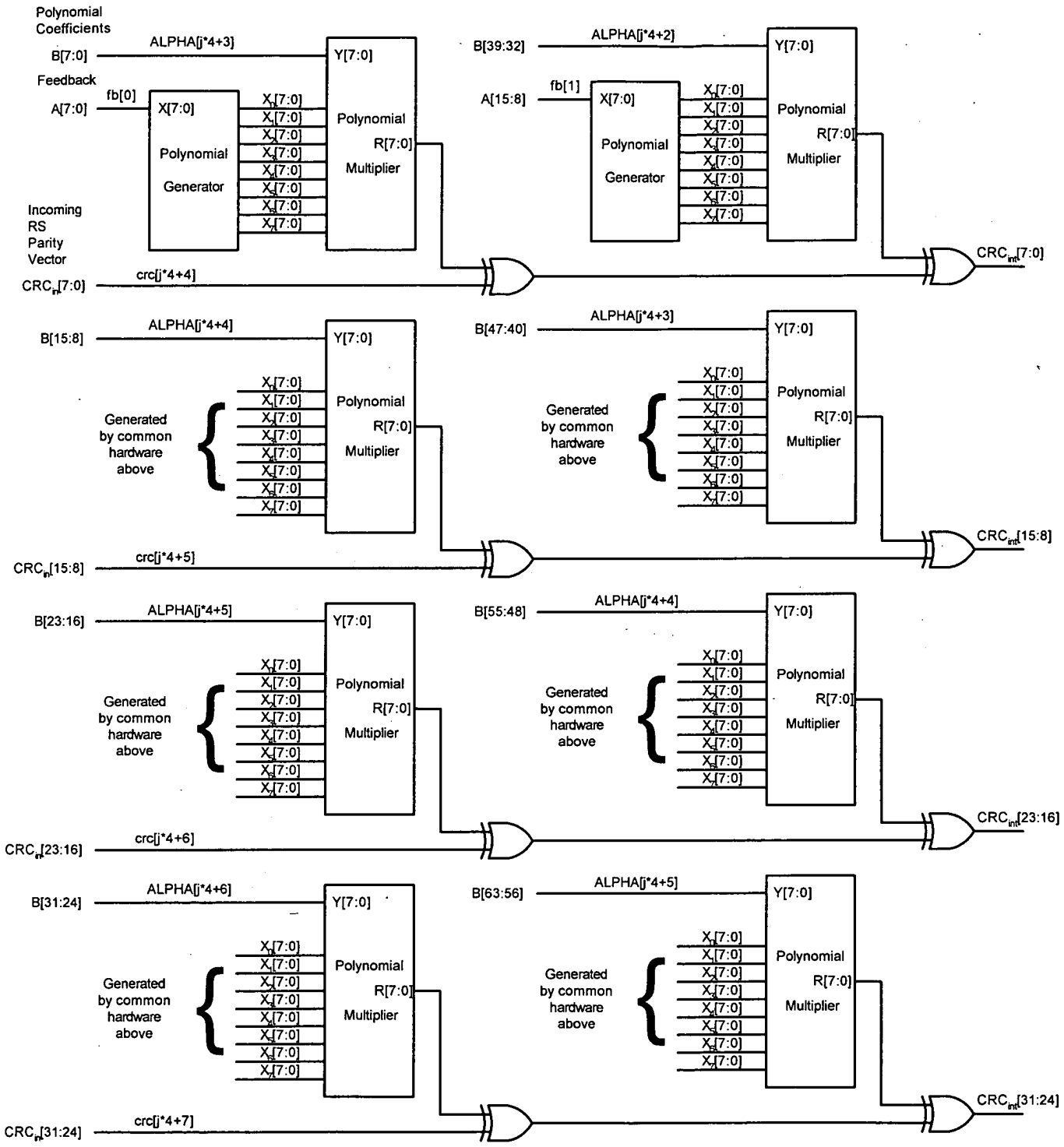
Figure 10



SIMD instruction: crc[3:0] = crc[3:0] ^ gf_mult (a, b[3:0])

As used in the example software, a is the byte feedback term and b is a set of four polynomial terms

Figure 11



GF Kernel instruction: crc[3:0] = crc[3:0] ^ gf_mult (a[3:0], b[15:0])

As used in the example software, a is a set of four byte feedback term and b is a set of sixteen polynomial terms

The set of sixteen polynomial terms could be referenced from a ROM as part of the GF Kernel instruction processor as only a small number of terms are necessary for each Reed Solomon coder type

Note: Byte indicies shown on instruction and bit indicies shown on hardware

Figure 12a

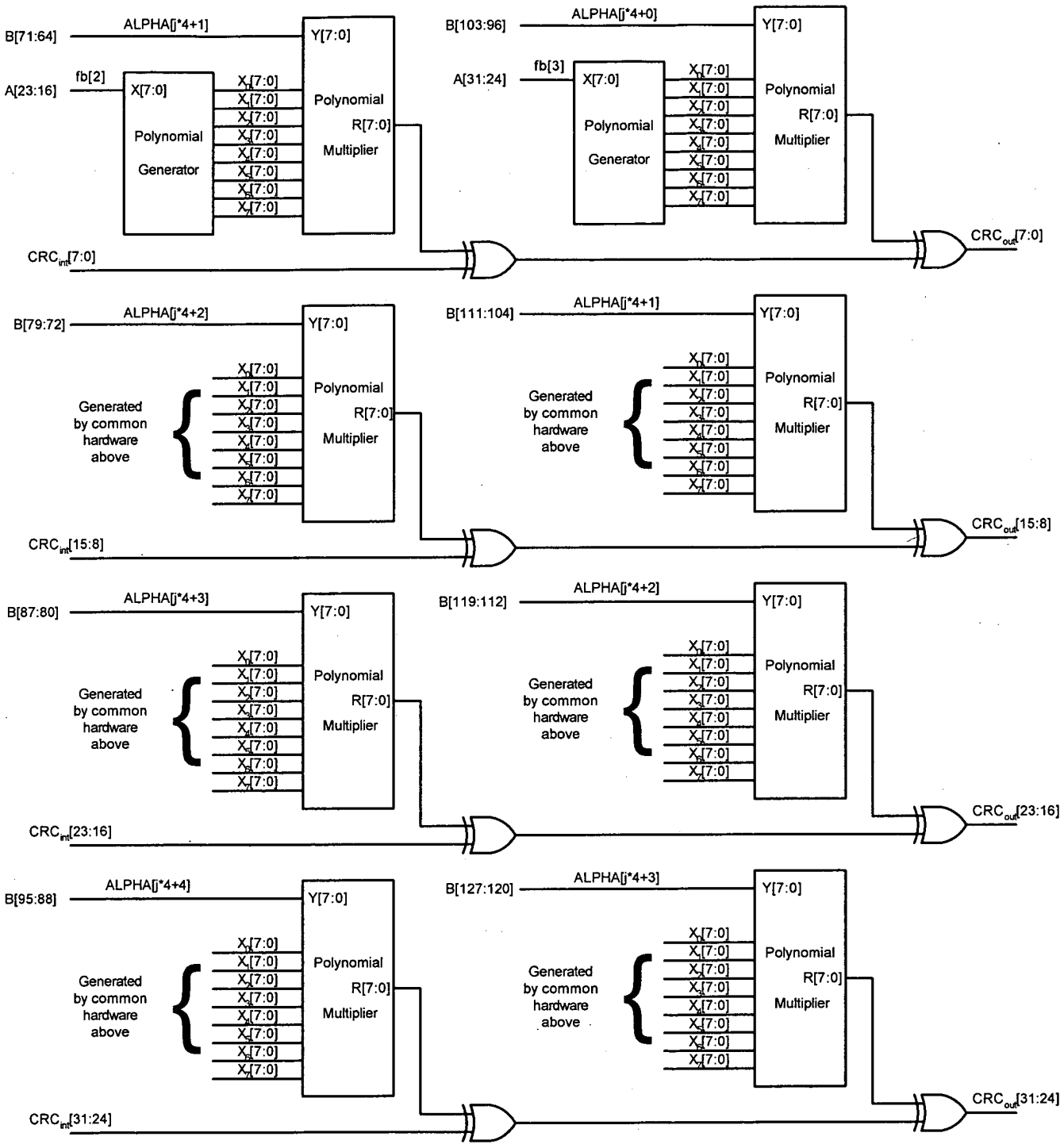
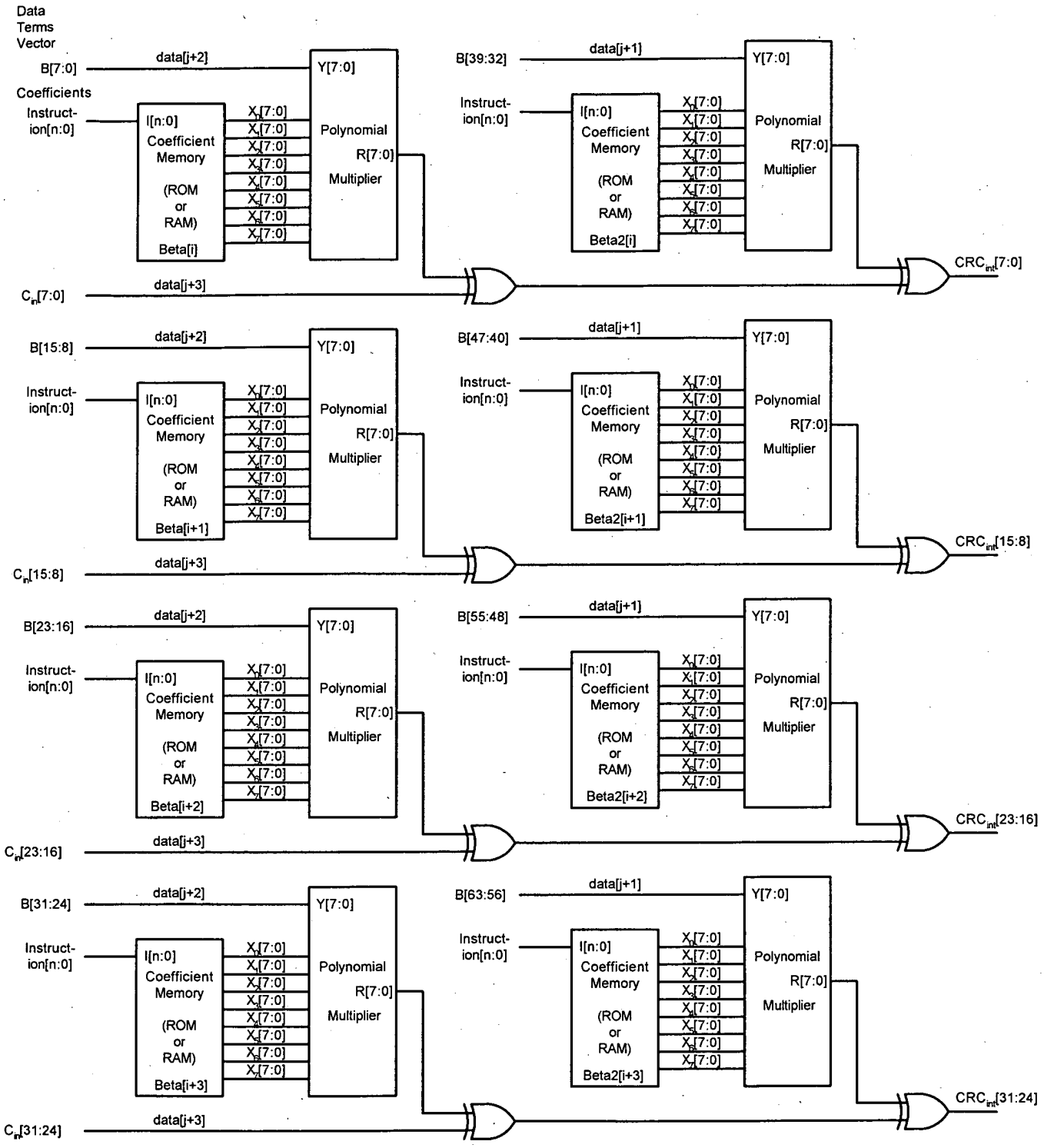

B[71:64]
ALPHA[j*4+1]
A[23:16]
fb[2]
X[7:0]
Polynomial
Generator
Y[7:0]
Polynomial
R[7:0]
Multiplier
B[103:96]
ALPHA[j*4+0]
A[31:24]
fb[3]
CRC_int[7:0]
CRC_out[7:0]
B[79:72]
ALPHA[j*4+2]
B[111:104]
ALPHA[j*4+1]
Generated
by common
hardware
above
CRC_int[15:8]
CRC_out[15:8]
B[87:80]
ALPHA[j*4+3]
B[119:112]
ALPHA[j*4+2]
CRC_int[23:16]
CRC_out[23:16]
B[95:88]
ALPHA[j*4+4]
B[127:120]
ALPHA[j*4+3]
CRC_int[31:24]
CRC_out[31:24]

Figure 12b



GF Kernel instruction: c[3:0] = c[3:0] ^ gf_mult (a[3:0], b[15:0])

As used in the example software, a is a set of four byte feedback term and b is a set of sixteen polynomial terms

The set of sixteen polynomial terms should be referenced from a ROM as part of the GF Kernel instruction processor as only a small number of terms are necessary for each Reed Solomon coder type

Note: Byte indicies shown on instruction and bit indicies shown on hardware

Figure 13a

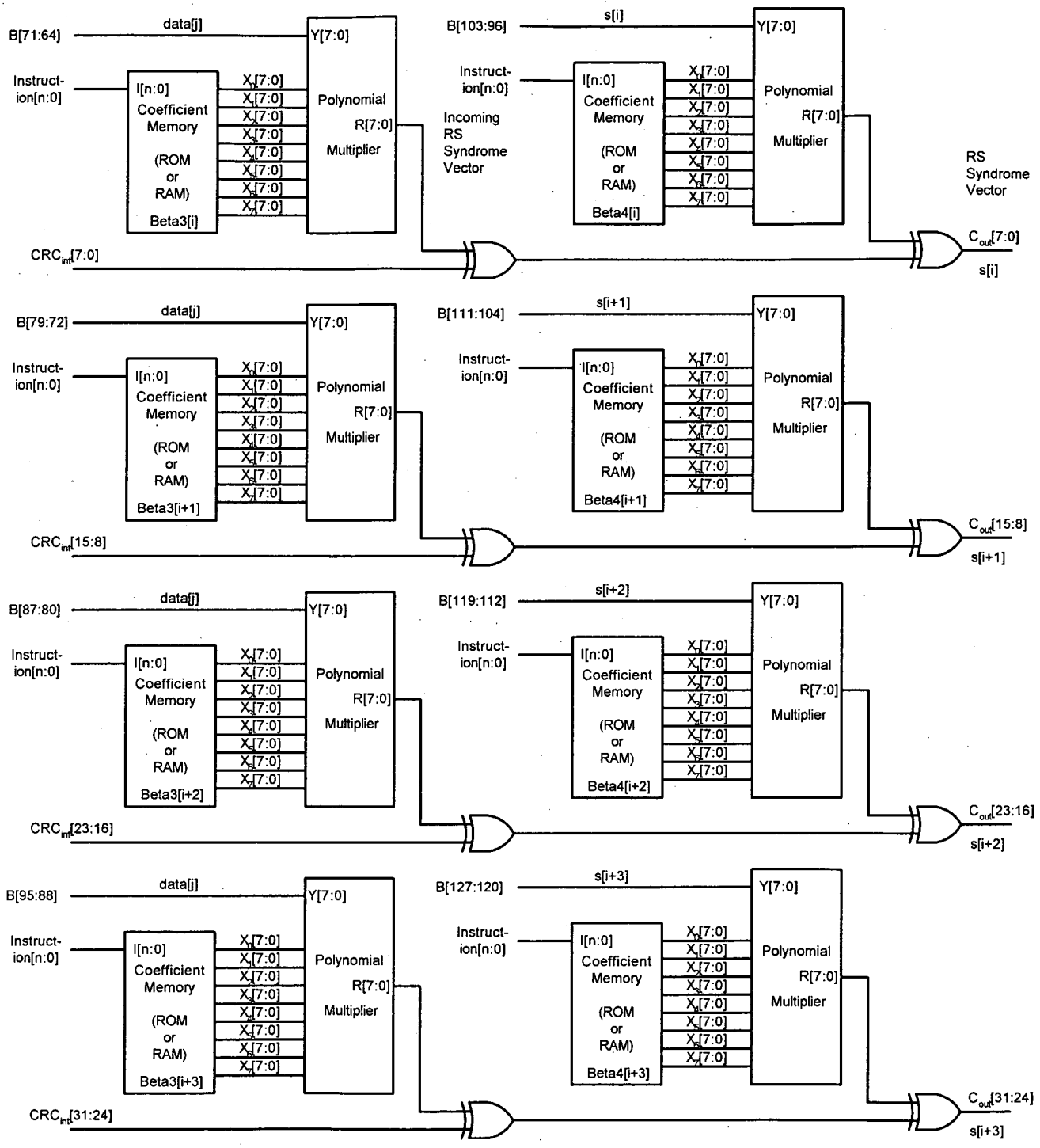
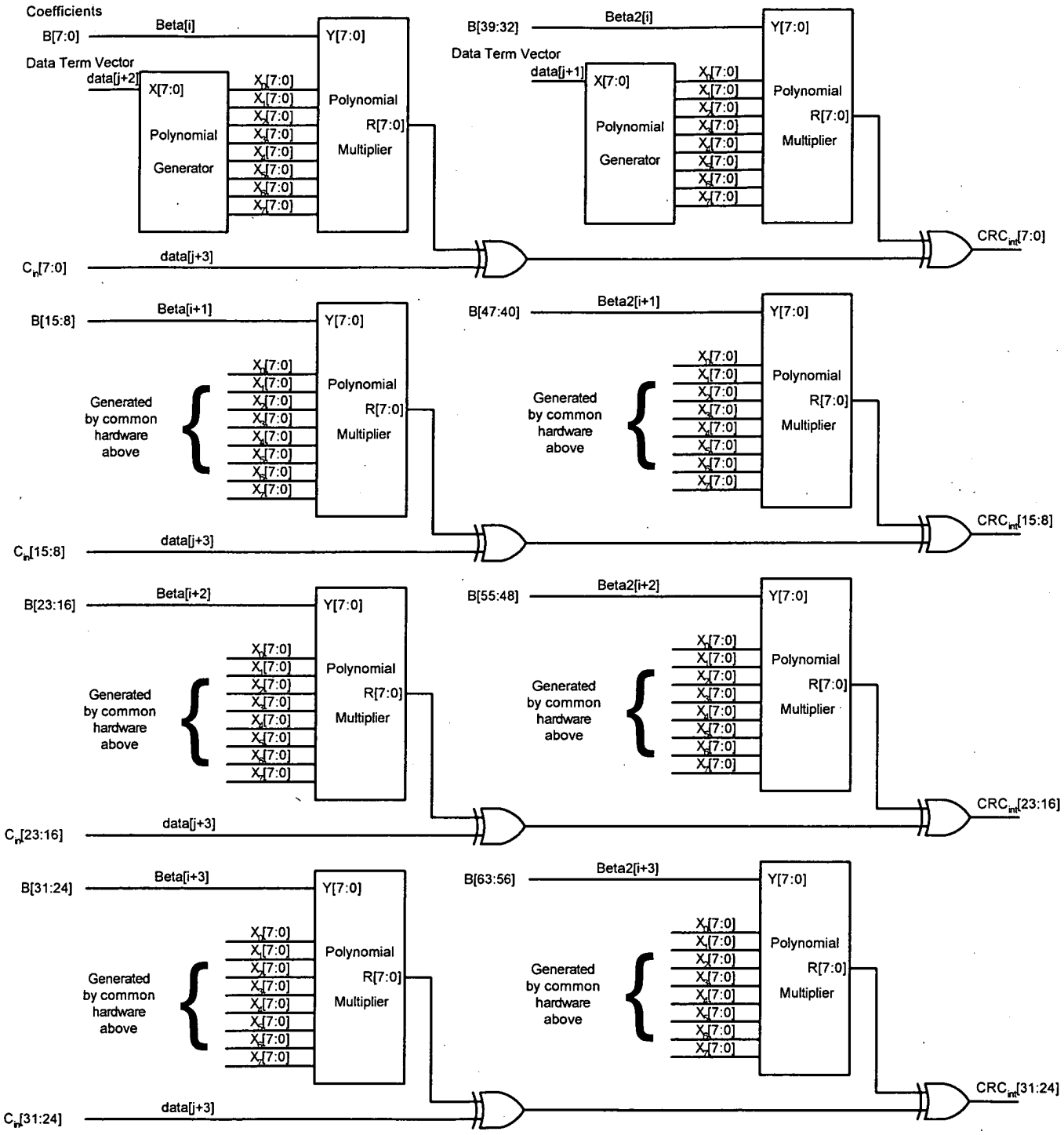

B[71:64]
data[j]
Y[7:0]
Instruct-ion[n:0]
I[n:0] Coefficient Memory (ROM or RAM) Beta3[i]
Polynomial Multiplier
R[7:0]
CRC_int[7:0]
B[103:96]
s[i]
Instruct-ion[n:0]
Incoming RS Syndrome Vector
I[n:0] Coefficient Memory (ROM or RAM) Beta4[i]
RS Syndrome Vector
C_out[7:0]
s[i]
B[79:72]
data[j]
Instruct-ion[n:0]
I[n:0] Coefficient Memory (ROM or RAM) Beta3[i+1]
CRC_int[15:8]
B[111:104]
s[i+1]
Instruct-ion[n:0]
I[n:0] Coefficient Memory (ROM or RAM) Beta4[i+1]
C_out[15:8]
s[i+1]
B[87:80]
data[j]
Instruct-ion[n:0]
I[n:0] Coefficient Memory (ROM or RAM) Beta3[i+2]
CRC_int[23:16]
B[119:112]
s[i+2]
Instruct-ion[n:0]
I[n:0] Coefficient Memory (ROM or RAM) Beta4[i+2]
C_out[23:16]
s[i+2]
B[95:88]
data[j]
Instruct-ion[n:0]
I[n:0] Coefficient Memory (ROM or RAM) Beta3[i+3]
CRC_int[31:24]
B[127:120]
s[i+3]
Instruct-ion[n:0]
I[n:0] Coefficient Memory (ROM or RAM) Beta4[i+3]
C_out[31:24]
s[i+3]

Figure 13b



GF Kernel instruction: c[3:0] = c[3:0] ^ gf_mult (a[3:0], b[15:0])

As used in the example software, a is a set of four byte feedback term and b is a set of sixteen polynomial terms

The set of sixteen polynomial terms should be referenced from a ROM as part of the GF Kernel instruction processor as only a small number of terms are necessary for each Reed Solomon coder type

Note: Byte indicies shown on instruction and bit indicies shown on hardware

Figure 14a

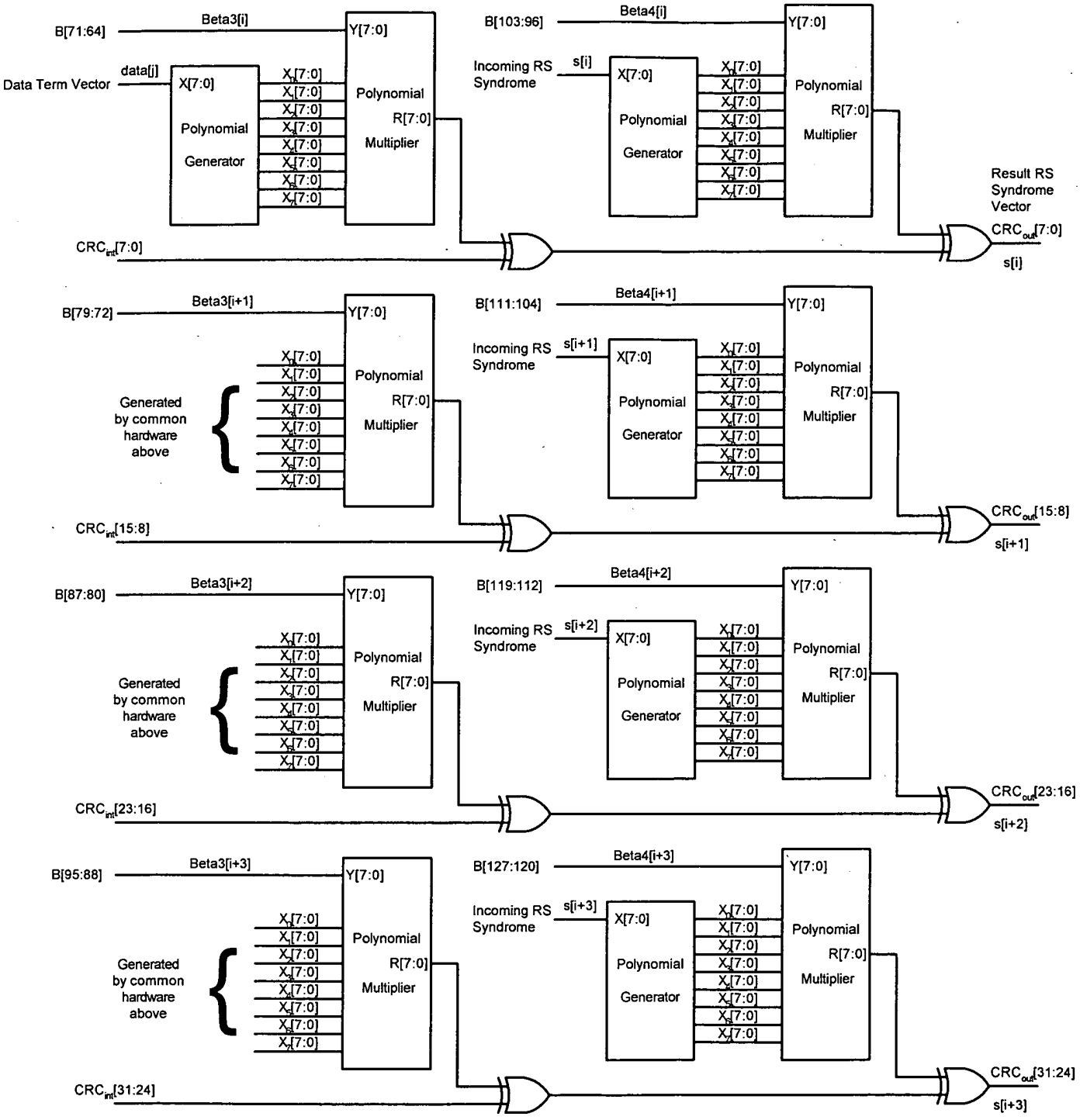

B[71:64]
Beta3[i]
Y[7:0]
Data Term Vector
data[j]
X[7:0]
Polynomial Generator
Polynomial R[7:0] Multiplier
B[103:96]
Beta4[i]
Incoming RS Syndrome
s[i]
Result RS Syndrome Vector
CRC_out[7:0]
s[i]
CRC_int[7:0]
B[79:72]
Beta3[i+1]
B[111:104]
Beta4[i+1]
Incoming RS Syndrome
s[i+1]
Generated by common hardware above
CRC_out[15:8]
s[i+1]
CRC_int[15:8]
B[87:80]
Beta3[i+2]
B[119:112]
Beta4[i+2]
Incoming RS Syndrome
s[i+2]
Generated by common hardware above
CRC_out[23:16]
s[i+2]
CRC_int[23:16]
B[95:88]
Beta3[i+3]
B[127:120]
Beta4[i+3]
Incoming RS Syndrome
s[i+3]
Generated by common hardware above
CRC_out[31:24]
s[i+3]
CRC_int[31:24]

Figure 14b